Simulating Gaze Attention Behaviors for Crowds

Journal:	*Computer Animation and Virtual Worlds*
Manuscript ID:	CAVW-09-0004
Wiley - Manuscript type:	Special Issue Paper
Date Submitted by the Author:	24-Mar-2009
Complete List of Authors:	Grillon, Helena; EPFL Thalmann, Daniel; EPFL
Keywords:	crowd animation, crowd realism, attention behaviors, crowd motion editing

Simulating Gaze Attention Behaviors for Crowds

Helena Grillon and Daniel Thalmann

Ecole Polytechnique Fédérale de Lausanne

IC ISIM VRLAB, Station 14

CH-1015 Lausanne, Switzerland

Tel. (+41)21 693 6646 Fax. (+41)21 693 5215

email: {helena.grillon,daniel.thalmann}@epfl.ch

Abstract

Crowd animation is a topic of high interest which offers many challenges. One of the most important is the trade-off between rich, realistic behaviors and computational costs. To this end, much effort has been put into creating variety in character representation and animation. Nevertheless, one aspect still lacking realism in virtual crowd characters resides in their attention behaviors. In this paper, we propose a framework to add gaze attention behaviors to crowd animations. First, we automatically extract interest points from character or object trajectories in pre-existing animations. For a given character, we assign a set of elementary scores based on parameters such as dis-

tance or speed to all other characters or objects in the scene. We then combine these subscores in an overall scoring function. The scores obtained from this function form a set of gaze constraints that determine where and when each character should look. We finally enforce these constraints with an optimized dedicated gaze Inverse Kinematics solver. It first computes the displacement maps for the constraints to be satisfied. It then smoothly propagates these displacements over an automatically defined number of frames. We demonstrate the efficiency of our method and our visually convincing results through various examples.

Keywords: crowd animation, crowd realism, attention behaviors, crowd motion editing

Introduction

When we walk in town, we look at other people, objects, or even at nothing in particular. An important aspect which can greatly enhance crowd animation realism is for characters to be aware of their environment and of other characters. This has partly been achieved with navigation and path planning. Our aim in this paper is to obtain more advanced behaviors than what navigation can provide. This raises the common problem of mandatory trade-off between rich, realistic behaviors and computational costs. To add attention behaviors to crowds, we are confronted to two issues. The first one is to detect the points of interest for characters to look at. The second one is to edit the character motions for them to perform the gaze behavior. This has to be done very rapidly in order to animate a large number of characters. In this paper, we propose a two-fold method which meets all these requirements.

Our first contribution is an automatic interest point detection algorithm based on *bottom-up* attention behaviors. These are passive or involuntary, stimulus-driven behaviors. Our algorithm automatically detects *where* and *when* each character should look. It is based on a scoring method which is a weighted sum of elementary scores. These are determined by functions using parameters such as distance or orientation. Our second contribution is a very fast dedicated Inverse Kinematics (IK) solver to satisfy these constraints. Our solver determines how the character motions are edited both spatially and temporally. It computes the displacement maps to satisfy the constraints and smoothly propagates the motion adjustments with adequate timing in order for the final motion to be fluid and continuous.

Related Work

Models of Human Vision and Perception. The synthesis of human vision and perception is a complex problem which has been tackled in many different ways. Models of synthetic vision based on memory have been developed for the navigation of characters [1, 2]. These models simulate vision but not the actual human gaze behavior. A model of perception was introduced by Hill [3] in which a character decided to attend to objects in an environment depeding on the information it received from them. Chopra Khullar and Badler [4] proposed an architecture which determined where an agent should look by selecting from top-down, bottom-up, and idling behaviors. However, their system requires that an animator insert the top-down interest points in a queue. Similarly, much work has been conducted in the simulation of visual attention and gaze in Embodied Conversational Agents [5, 6, 7]. These models give very convincing results but are not applicable to crowds. Several researchers proposed perceptual systems based on saliency maps [8, 9, 10]. Kim et al. [11] expanded the approach by using a benefit and cost function to determine when a character should look at an object. The saliency-map method gives very good results but is prohibitive for crowd animation. Yu and Terzopoulos [12] proposed a decision network framework to simulate how people make decisions on what to attend to and how to react. Their system, however, is aimed at simulating situations with a small group of people.

Motion Editing. A large category of motion editing methods relies on the skillful manipulation of clips from a motion capture database [13, 14, 15, 16]. Due to the many possible

configurations in attention behaviors, this would require a very dense database. Other methods used analytic IK [17, 18]. Lee and Shin [19] proposed a method to edit a pre-existing animation to satisfy a set of user defined constraints. Shin et al. [20] used Kalman filters and a set of rules to assign varying importance to a set of tasks which they then solved with a dedicated IK solver. Kulpa et al. [21] proposed a hierarchical Cyclic Coordinate Descent algorithm to deal with spacetime constraints. These analytic methods are dedicated to the positioning of end-effectors, whereas we are interested in controlling the final orientation of the eyes, head, and spinal joints over time. Several other methods used Jacobian based IK solvers to edit motions. For example, Choi and Ko [22] discussed a method for online retargetting. Le Callennec and Boulic [23] introduced the notion of prioritized constraints to solve possible conflicts between user-defined constraints. While these methods are generic enough to possibly use any kind of constraints, the use of Jacobian inversion causes prohibitive computational costs that are not compatible with our framework.

On a different note, Lee at al. [24] described an eye movement model based on statistical and saccade empirical models of eye-tracking data. Lee and Terzopoulos [25] proposed a head-neck model based on biomechanics. These methods give stunning results but once again are not applicable to crowds.

System Overview

Our system works as an extra layer added to an existing crowd animation. We enhance this animation by providing its characters with gaze behaviors. For clarity purposes, we use the term *character* to refer to the individual for which we are generating the gaze behavior and the term *entity* to refer to either a character or an object that can possibly attract attention. Finally, *interest points* are the locations which attract attention. Our method generates gaze behaviors solely from the entities' trajectories. Thus, it is generic, and can be used with any type of crowd animation engine. We define a trajectory $\mathbf{T}_i(t)$ for an *entity* E as:

$$\mathbf{T}_i(t) = [\mathbf{p}_i(t), r_i(t)] \tag{1}$$

where i is the entity's ID, $\mathbf{p}_i(t) \in \mathbb{R}^3$ its position at time t, and $r_i(t) \in \mathbb{R}$ its forward orientation at time t. Since our method aims at enhancing crowd realism, we must deal with a large number of characters. It would be unthinkable for a user to define all the points of interest to be attended to by each character. It is thus mandatory to automatically detect them. This is one of the key features of our method. Based on the entity input trajectories, it takes into account both the spatial and the temporal aspects of gaze behaviors. Finally, the detected interest points form a set of gaze constraints $\mathbf{L}$ to be satisfied.

Our method also consists of a dedicated IK solver. Given an existing motion, we compute the displacement maps $\mathbf{m}(t_i)$ that adjust the postures in order to satisfy the automati-

cally detected constraints. We then propagate these $\mathbf{m}(t_i)$ in order for the eyes, head, and torso to be desynchronized, i.e. the movement is initiated by the eyes; the head and the torso then follow and the eyes partially recenter with respect to the head. The character motions are thus adapted for them to attend to the interest points in a smooth and natural way.

Automatic Interest Point Detection

The first step in our method consists of automatically detecting the interest points from the entity trajectories. We define an interest point IP as an entity E which should be attended to by a given character C. More formally, IP is defined as:

$$IP(t) = [\mathbf{p}_t, t_a, t_d, [t_b, t_e]] \;\; where \;\; \mathbf{p}_t \in \mathbb{R}^3 \tag{2}$$

where $\mathbf{p}_t$ is IP's position in space at time t. t_a is its *activation duration*, t_d its *deactivation duration*, and $[t_b, t_e]$ represents its lifespan. The purpose of t_a is to define the amount of time it will take for the looking motion to be executed. Conversely, t_d defines the amount of time for C to look away from IP. These are further discussed in a later section of this paper. It is to be noted that in the case where the IP is replaced by another, the deactivation is skipped and replaced by the activation to go from the first IP to the second.

Another important factor in gaze behaviors is that we do not look at things indefinitely. We can either loose interest or find something else more interesting to look at. As shown

in Figure 1, we regulate this with $[t_b, t_e]$. It is the duration for which an entity E *is* an IP. For each character C and at each time t, we define the level of interest other entities have by assigning them a score $S(t)$ computed through a scoring function. The entity E which obtains the highest score $S_{max}(t)$ becomes the IP that should be attended to by C at time t as long as it fulfills two conditions. $S_{max}(t)$ first has to be above an *attention threshold*. This defines the percentage of time C will be attentive to other entities. Second, E should obtain $S_{max}(t)$ for a minimal amount of time $[t_b, t_e]$ which we have empirically set to $1/3s$.

Previous studies such as Neisser's [26] explain that human attention is captured by substantial differences in one or more simple visual attributes. Simple visual attributes are features such as color, orientation, size, and motion [27]. Additionally, Yantis and Jonides [28] underlined that abrupt visual onsets equally attract human attention. These studies have motivated our choice of four different criteria as components to our scoring function:

Proximity: closer objects or people seem larger and attract attention more easily than those far away. Moreover, those which are closer occlude those which are further away.

Relative speed: a person will be more prone to set his/her attention on something moving fast than moving slowly relative to his/her own velocity.

Relative orientation: we are more attentive to objects coming towards us than moving away from us. Moreover, something coming towards us seems to become larger.

Periphery: we are very sensitive to movements occurring in the peripheral vision. More specifically, to objects or people entering the field of view.

To decide where a given character will look at a given time we evaluate all entities in

terms of these criteria. As depicted in Figure 2, we evaluate a set of parameters for each of these entities: the distance $d_{ce}(t)$, the relative speed $rs(t)$ defined by forward differentiation as $||\mathbf{d}_e(t) - \mathbf{d}_c(t)||$, the orientation in the field of view $\alpha(t)$, and the relative direction $\beta(t)$. Similarly to Sung et al. [29], we then combine these parameters to create more complex scoring functions: S_p for proximity, S_s for speed, S_o for orientation, and S_{pe} for periphery.

The *proximity* parameter evaluates the distance between a character C and all other entities E. Given $d_{ce}(t)$, the distance between C and E at time t, and $\alpha(t)$ the orientation of E in C's field of view at time t, our proximity score is computed as:

$$S_p(t) = \exp(\frac{-(0.5(d_m - d_{ce}(t)) + (\frac{d_m}{2} - 1))^2}{2}) \quad (3)$$

where d_m is the maximal distance value beyond which C will stop looking. We allow for entities situated $2 - 3m$ away from C to obtain the highest scores. We believe those closer than this will already have been attended to and should loose their interest potential.

For *speed*, we follow the same principle as for proximity. It is computed as:

$$S_s(t) = \omega_{sw}||\mathbf{d}_e(t) - \mathbf{d}_c(t)|| \quad (4)$$

where $||\mathbf{d}_e(t) - \mathbf{d}_c(t)||$ is the relative speed and ω_{sw} is an arbitrary weighting factor to bring the speed scores to vary in the same range as the proximity ones. $S_s(t)$ expresses the difference between the distances traveled by E and C in one frame.

Similarly, our *orientation* score is computed as:

$$S_o(t) = (\pi - \alpha(t))\beta(t) \tag{5}$$

The larger $\alpha(t)$, the more opposite the directions of E and C will be. We want to give more importance to the entities coming towards C. We thus weight the score in order for the entities in the central vision to be favored as opposed to the entities in the peripheral vision.

The last criterion is *periphery*. The calculations are the same as for the orientation, however, we give more importance to the entities in the periphery. Its score is computed as:

$$S_{pe}(t) = \begin{cases} 0 & \text{if } \beta(t) > \beta_m \\ \omega_{pw}\alpha(t)(\pi - \beta(t)) & \text{otherwise} \end{cases} \tag{6}$$

where β_m is the maximum angle between the forward directions of C and E. Here as well, we weight the score with a weighting parameter ω_{pw} for the score range to be similar to that of the other criteria. We thus obtain all our subscores.

It is important to note that we further improve our algorithm by pruning a number of computations. First, we use the maximum distance d_m. All entities farther than this from C are automatically discarded from further computation. Out of this subset, we prune the process again by considering only the entities in C's field of view. All following computations are done on this remaining subset of entities. We thus greatly reduce computational costs.

At this point, we can define, for each parameter individually, which entity is the most

interesting for each character at each frame. However, these criteria need to be evaluated as a whole for them to have a meaning. To this end, we define a final scoring function. Moreover, in order to obtain variety in the character gaze behaviors, we want to bring in subtle changes in the importance of each parameter. The subscores can thus be weighted to have more or less influence on the overall score. These weights are randomly assigned by the application and sum up to 1. Our overall scoring function is thus defined as:

$$S(t) = I_E(\omega_p S_p(t) + \omega_s S_s(t) + \omega_o S_o(t) + \omega_{pe} S_{pe}(t)) \tag{7}$$

where I_E is the *impact factor* of E. Once the best overall scores $S_{max}(t)$ have been computed, we define the attention threshold A which determines the minimum score for a gaze behavior to be activated. This cannot be defined as an absolute value since the overall scores can greatly vary. We thus compute A as the *(100-a)*th percentile of $S_{max}(t)$. C will thus only pay attention a percent of the time. As depicted in Figure 1, the gazed at IPs will be the ones that have a higher value than A. We thus partially simulate mood or personality.

Our method automatically generates gaze shifts since we calculate the IPs at each frame and for each character. However, if the IP stays the same for a long time, this generates unlikely behaviors. For example, if two characters are walking side by side, their respective scores for each other may be very high due to their proximity. They will thus keep on staring at each other, producing unrealistic gaze behaviors. We therefore define a threshold duration d_l. If an IP lasts for more than d_l, the entity of next highest interest is chosen as

new IP. We empirically set d_l to a maximum value of 4 seconds for the gaze behavior not to last indefinitely. This also simulates interesting emergent behaviors. In the example given above, two characters walking side by side will oscillate between looking at each other and looking at another entity or back in front of them. They will thus seem to be talking together.

Automatic Motion Adaptation for Gaze

In the present section, we explain how we adapt the initial motions to obtain the desired gaze behaviors. Each of the IPs we have calculated for a character C can be considered as a gaze constraint l_i in a set of gaze constraints $\mathbf{L}$. C's motion thus has to be adjusted to meet these l_i. Since the IPs can be dynamic (in the case where they are moving entities), we have to compute the joint displacements to be applied to the base motion at each frame. As this is done on a per-frame basis, the overall performance of our system critically depends on our IK solver. To this end, we propose a robust and very fast dedicated IK solver.

The skeletons we use are composed of 86 joints. Our method adjusts 10 of them: 5 spinal cord, 2 cervical, 1 head and 2 eye joints, in order for the characters to align their gaze to the IPs. The eyes are *swing* joints and have 2 degrees of freedom (DOF). All the others are *ball and socket* joints that have 3 DOF. This amounts to 28 DOF in all. By considering only this subset of the full skeleton, we greatly reduce the complexity of our algorithm. This allows us to have very small computational times and thus to animate a large number of characters.

Our method consists of two distinct phases. The first one computes the displacement

map to be applied in order to satisfy the current gaze constraint. We name this *spatial resolution*. At each timestep, if there is an active constraint, we launch an iterative loop starting with the bottom of the kinematic chain (lumbar vertebras) and ending with the top of the kinematic chain (eyes). At each iteration, we calculate the total remaining rotation to be done by the average eyes position (global eye) to satisfy the constraint and determine the ratio of this rotation to be applied to the current joint. The remaining rotation to be done by each eye joint is then computed in order for them to converge on the IP. Moreover, for IPs in the $30°$ composing the central foveal area, only the eye joints are recruited. For the $15°$ farther on each side composing the central vision area, only the eye, head, and cervical joints are recruited. Small movements therefore do not recruit the heavier joints. Similarly, for larger movements, the final $15°$ are done by the eyes only and the $15°$ before that, by the eyes, head and cervicals only. The second component is the *temporal* propagation of the displacement map over an automatically defined number of frames. This number is different if considering the eyes, the head and cervicals, or the joints composing the remainder of the spine. In this way, we allow for the lighter joints to move more rapidly than the others. The eyes thus converge on the IP well before any of the other joints attain their final posture.

Spatial Resolution

The purpose of the spatial resolution is to find a *displacement map* $\mathbf{m}(t)$ that modifies the initial motion in order to satisfy a given gaze constraint l_i. Similarly to Lee and Shin [19],

we consider the initial motion as a set of independent character postures. We adjust each of these postures individually to satisfy the constraint. To determine the displacement which should be applied to each of the recruited joints, we first calculate the 3D rotation $\mathbf{q}_l \in \mathbb{S}^3$, that aligns the global eye orientation to the position of l_i. Let $\mathbf{M}_{wt}$ be the rigid transformation matrix that transforms a point $\mathbf{p}$ in a local coordinate frame to its world position $\mathbf{x}_{wt}$ at time t. Let $\mathbf{l}_{wt}$ be IP's position expressed in world coordinates. The vector $\mathbf{v}_{lt}$ going from the global eye to the IP in the global eye frame is defined as:

$$\mathbf{v}_{lt} = \mathbf{R}_{wt}^T(\mathbf{l}_{wt} - \mathbf{x}_{wt}) \tag{8}$$

where $\mathbf{R}_{wt}$ is the rotational part of $\mathbf{M}_{wt}$ and $\mathbf{x}_{wt}$ is the global eye position in the world coordinate frame. Let $\mathbf{d}_{lt}$ be the initial looking direction expressed in the global eye frame. The total rotation $\mathbf{q}_{lt}$ in local coordinates is thus the shortest rotation to go from $\mathbf{d}_{lt}$ to $\mathbf{v}_{lt}$. The eyes are not the only joints to adjust. To reach a natural posture, we dispatch this rotation to the other recruited joints. To determine the contribution c_i of each joint to the complete rotation $\mathbf{q}_l$, we take inspiration from Boulic et al. [30]. We use the formula they propose for the spinal rotation distribution around the vertical axis. In our model, the rotations around the other axes are very small; we therefore keep the same formula for all types of rotations:

$$c_i = (-(i-n))(\frac{2}{n(n-1)}) \quad i = 1...9 \tag{9}$$

where n is the total number of joints through which to iterate and i is the joint index. At

each step, c_i determines the percentage of *remaining* rotation to be assigned to joint i. The total rotation to be done by each joint for the character to satisfy the constraint may then be calculated by spherical linear interpolation using these contribution values. To reach the final posture, we compute the remaining rotation for each eye to converge on the IP.

Temporal Resolution

The speed of our looking motions varies depending on what we look at. To reproduce this, we dynamically determine the activation duration based on the best overall scores $S_{max}(t)$. A point of high interest triggers a rapid movement and one of low interest a slower one. The activation duration t_a for a character C to satisfy a constraint l_i is thus computed with the $S_{max}(t)$ at time t_b associated to that constraint. Given the maximum possible score S_{MAX}, t_a is computed as:

$$t_a = \frac{\alpha S_{MAX}}{v_m S_{max}(t)} \tag{10}$$

where α is the angle of the total rotation which would have to be done by the head to satisfy l_i, expressed in radians and v_m is the maximum possible head velocity. The choice of value for v_m is motivated by a study conducted by Grossman et al. [31]. The authors experimented on the maximum head velocity during vigorous voluntary yaw rotations. They obtained a median maximal velocity v_m of 4π rad/s. However, we hardly use our maximal head velocity. We therefore set it to 2π rad/s in our model. t_a defines the number of frames it

will take the head and cervical joints to satisfy l_i. We double this value to obtain the number of frames in which the remainder of the spine will satisfy l_i and halve it to obtain the number of frames in which the eyes will converge. This allows for the lighter joints to move faster than the heavier ones. Finally, since a motion can take as long as one wants, there is no particular time threshold under which it should be done. We therefore empirically set an upper bound value for t_a at 2 seconds in order for the motion not to be unnaturally long. In this way, if $S_{max}(t)$ is very high, the turning motion will be done fast and if it is low, the turning motion will be done in a larger number of frames. Moreover, the eyes converge on the IP faster than the head and cervical joints, which in turn will satisfy the constraint before the remainder of the spine. If the gaze behavior is deactivated, either because C has been looking at an IP for too long or if there are no more IPs above the attention threshold, C will look back in front of him/her, i.e, will return to its original posture. The duration of the deactivation t_d is randomly generated within an adequate range.

Our gaze movements are not performed at a linear velocity. They start with an acceleration or ease-in phase, reach a peak velocity, and end with a deceleration or ease-out phase [25]. They are also desynchronized in time. The eyes move before the head, which moves before the torso. To reproduce this, we further weight our rotation contributions c_i with a temporal propagation function $f_P(t)$ which follows a Gauss error function curve:

$$f_P(t) = erf(n/2) = \frac{2}{\sqrt{\pi}} \int_{-n/2}^{n/2} e^{-t^2} \tag{11}$$

where n is the number of frames over which the gaze motion will be done. This computation is done with the different activation time values for the three sets of joints (eyes, head and cervicals, and torso). As depicted in Figure 3, we thus obtain a slight delay in the movement initiation between these three sets of joints. Our final movement therefore allows for the eyes to converge on the IP and then partially recenter with respect to the head as the remainder of the joints move to satisfy l_i. In our examples, most characters are in movement and the majority of the constraints are associated to other entities in movement. These constraints are thus *dynamic*. We therefore recompute the displacement map to satisfy l_i at each timestep. We can assume that its position from one frame to the next one does not change much. We therefore recompute the rotation to be done at each frame but maintain the total contribution $f_P(t)c_i$ to apply which we calculated before the initiation of the gaze motion. However, we reset the contributions to 0 if the gaze constraint changes, i.e., if it is associated to another entity situated elsewhere in the scene. More specifically, it is the case when the current constraint location is farther than a pre-determined threshold from the constraint location at the previous frame. The newly calculated rotations to attain the new constraint position are then distributed over the appropriate number of frames.

Experimental Results

We used our framework to create examples of the possibilities of our method. The motion clips for our examples have been sampled at 30 fps. All the animations were generated on

an Intel Core 2 Duo 3.0 GHz, 2GB RAM and an NVidia GeForce 8800 GT graphics board. For these examples, we have used the crowd simulation engine described in [32].

Our first example illustrates the desynchronization between the three sets of joints and various parameters by applying them individually to a single character C. Figure 4 depicts the desynchronization and the periphery parameter. On the left, C's eyes converge on the IP while the head and spine have not yet satisfied the constraint. On the right, maximal values have been set to the periphery and attention parameters. The maximum looking duration is not activated in this example since it aims at demonstrating solely the motion editing.

In our second example, we illustrate the use of our scoring algorithm together with the motion editing over 130 characters walking up and down a street, standing, or sitting on a bench. This is depicted in Figure 5. The maximum distance threshold was set to $10m$. The attention threshold and the importance of each parameter was randomly generated by our application and is different for each character. For each one, the scoring algorithm is applied to all other eligible entities. Additionally, it is applied to all eligible scene objects defined as potential IPs (60 in all). We can thus simulate a simple form of top-down attention in the sense that some characters seem to be looking for something or trying to find their way. An interesting aspect emerging from those results is that some characters walking or standing next to each other regularly look at each other. They seem to be talking to each other.

Concerning complexity and computational times, our automatic IP detection algorithm is in $\mathcal{O}(n^2)$ with n being the number of characters. Indeed, for each character C, we have to evaluate all other entities E. However, since we do not compute the IPs for entities out

of C's field of view and farther than a distance threshold from it, this is greatly reduced and depends on population density. For the last example, the computational time for the IP detection, per character and per frame, was of $0.036ms$. We have also tested our IP detection's computational times. These are expressed in milliseconds, per character and per frame. For 100 characters, the computational time was of $0.017ms$, for 200, it was of $0.033ms$, for 500, it was of $0.088ms$, and for 1000, it was of $0.177ms$. The automatic IP detection can thus be done for hundreds of characters in real-time. However, the computational times for 1000 characters is prohibitive. Nevertheless, this is hardly necessary since users would only perceive those behaviors in the foreground. We have also compared our IK solver with a typical Jacobian-based IK approach [33]. To perform this comparison, the same skeleton has been used in both cases. We then placed an IP in several different locations. On average, the Jacobian-based approach took $20ms$ per iteration to satisfy the constraint. This method needs about 15 iterations before converging, which amounts to $300ms$ to solve a constraint. Our method took a mean time of $0.3ms$ to solve a constraint. Since it is an analytical approach, we do not need more than 1 iteration to solve it. The complexity of our IK solver is therefore in $\mathcal{O}(n)$ with n being the number of characters.

Conclusion

In this paper, we introduced a novel method to enhance crowd animation realism by adding attention behaviors to the characters composing it. We first proposed an automatic interest

point detection algorithm which determines, for each character, *where* and *when* it should look. We additionally presented an extensible and flexible set of criteria to determine interest points in a scene and a method to combine them. Our method also allows the fine-tuning of character attention behaviors by introducing an attention parameter as well as the possibility to modify the relative importance of each criterion if desired. As a second contribution, we introduced a robust and very fast dedicated gaze IK solver to edit the character motions. Our solver deals with the spatial and temporal resolution of the gaze constraints defined by our detection algorithm. Finally, we illustrated our method with visually convincing results obtained with the combination of both our contributions.

Acknowledgements

The authors would like to thank Mireille Clavien for the design. They would also like to thank Benoît Le Callennec, Ronan Boulic, Daniel Raunhardt, Anders Sandholm and Barbara Yersin for careful proofreading.

References

[1] J.J. Kuffner, Jr, and J.-C. Latombe. Fast synthetic vision, memory, and learning models for virtual humans. In *Proceedings of Computer Animation*, pages 118–127, 1999.

[2] C. Peters and C. O'Sullivan. Synthetic vision and memory for autonomous virtual humans. *Computer Graphics Forum*, 21(4):743–752, 2002.

[3] R. Hill. Modeling perceptual attention in virtual humans. In *Proceedings of Computer Generated Forces and Behavioral Representation*, 1999.

[4] S. Chopra Khullar and N.I. Badler. Where to look? automating attending behaviors of virtual human characters. *Autonomous Agents and Multi-Agent Systems*, 4(1-2):9–23, 2001.

[5] M. Gillies. *Practical Behavioural Animation Based On Vision And Attention*. PhD thesis, University of Cambridge, 2001.

[6] C. Peters, C. Pelachaud, E. Bevacqua, M. Mancini, and I. Poggi. A model of attention and interest using gaze behavior. *Lecture Notes in Computer Science*, 3661:229–240, 2005.

[7] E. Gu and N. Badler. Visual attention and eye gaze during multiparty conversations with distractors. *Lecture Notes in Computer Science*, 4133:193–204, 2006.

[8] L. Itti, N. Dhavale, and F. Pighin. Realistic avatar eye and head animation using a neurobiological model of visual attention. In *Proceedings of the Symposium on Optical Science and Technology*, volume 5200, pages 64–78, August 2003.

[9] C. Peters and C. O'Sullivan. Bottom-up visual attention for virtual human animation. In *Proceedings of Computer Animation and Social Agents*, pages 111–117, 2003.

[10] E. Marchand and N. Courty. Controlling a camera in a virtual environment. *The Visual Computer*, 18(1):1–19, 2002.

[11] Y. Kim, R.W. Hill, Jr, and D.R. Traum. A computational model of dynamic perceptual attention for virtual humans. In *Proceedings of Behavior Representation in Modeling and Simulation*, 2005.

[12] Q. Yu and D. Terzopoulos. A decision network framework for the behavioral animation of virtual humans. In *SCA '07: Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 119–128, 2007.

[13] L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. In *Proceedings of ACM SIGGRAPH, Annual Conference Series*, pages 473–482, 2002.

[14] O. Arikan and D. Forsyth. Interactive motion generation from examples. In *Proceedings of ACM SIGGRAPH, Annual Conference Series*, pages 483–490, 2002.

[15] J. Lee, J. Chai, P.S.A. Reitsma, J.K. Hodgins, and N.S. Pollard. Interactive Contol of Avatars Animated With Human Motion Data. In *Proceedings of ACM SIGGRAPH, Annual Conference Series*, pages 491–500, jul 2002.

[16] L. Kovar and M. Gleicher. Flexible automatic motion blending with registration curves. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 214–224, 2003.

[17] N.I. Badler, J.D. Korein, J.U. Korein, G.M. Radack, and L. Shapiro Brotman. Positioning and animating human figures in a task-oriented environment. *The Visual Computer*, 1(4):212–220, 1985.

[18] D. Tolani, A. Goswami, and N.I. Badler. Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical models*, 62(5):353–388, 2000.

[19] J. Lee and S.Y. Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of ACM SIGGRAPH, Annual Conference Series*, pages 39–48, 1999.

[20] H.J. Shin, J. Lee, S.Y. Shin, and M. Gleicher. Computer puppetry: An importance-based approach. *ACM Transactions on Graphics*, 20:67–94, 2001.

[21] R. Kulpa, F. Multon, and B. Arnaldi. Morphology-independent representation of motions for interactive human-like animation. In *EURORAPHICS 2005*, volume 24, pages 343–352, 2005.

[22] K.-J. Choi and H.-S. Ko. Online motion retargetting. *The Journal of Visualization and Computer Animation*, 11(5):223–235, 2000.

[23] B. Le Callennec and R. Boulic. Interactive motion deformation with prioritized constraints. In *Proceedings of the ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 163–171, 2004.

[24] S.P. Lee, J.B. Badler, and N.I. Badler. Eyes alive. In *Proceedings of ACM SIGGRAPH, Annual Conference Series*, pages 637–644, 2002.

[25] S.-H. Lee and D. Terzopoulos. Heads up!: biomechanical modeling and neuromuscular control of the neck. In *Proceedings of ACM SIGGRAPH, Annual Conference Series*, pages 1188–1198, 2006.

[26] U. Neisser. *Cognitive psychology*. Appleton-Century-Crofts, New York, USA, 1967.

[27] J.M. Wolfe and T.S. Horowitz. What attributes guide the deployment of visual attention and how do they do it? *Nature Reviews Neuroscience*, 5(6):495–501, 2004.

[28] S. Yantis and J. Jonides. Abrupt visual onsets and selective attention: voluntary versus automatic allocation. *Journal of Experimental Psychology: Human Perception and Performance*, 16(1):121–134, 1990.

[29] M. Sung, M. Gleicher, and S. Chenney. Scalable behaviors for crowd simulation. *Computer Graphics Forum*, 23(3):519–528, 2004.

[30] R. Boulic, B. Ulicny, and D. Thalmann. Versatile walk engine. *Journal of Game Development*, 1(1):29–43, 2004.

[31] G.E. Grossman, R.J. Leigh, L.A. Abel, D.J. Lanska, and S.E. Thurston. Frequency and velocity of rotational head perturbations during locomotion. *Experimental Brain Research*, 70(3):470–476, 1988.

[32] J. Maïm, B. Yersin, J. Pettré, and D. Thalmann. Yaq: An architecture for real-time navigation and rendering. *IEEE Computer Graphics & Applications Special Issue on Virtual Populace*, in Press, 2009.

[33] M. Peinado, D. Meziat, D. Maupu, D. Raunhardt, D. Thalmann, and R. Boulic. Accurate on-line avatar control with collision anticipation. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 89–97, 2007.

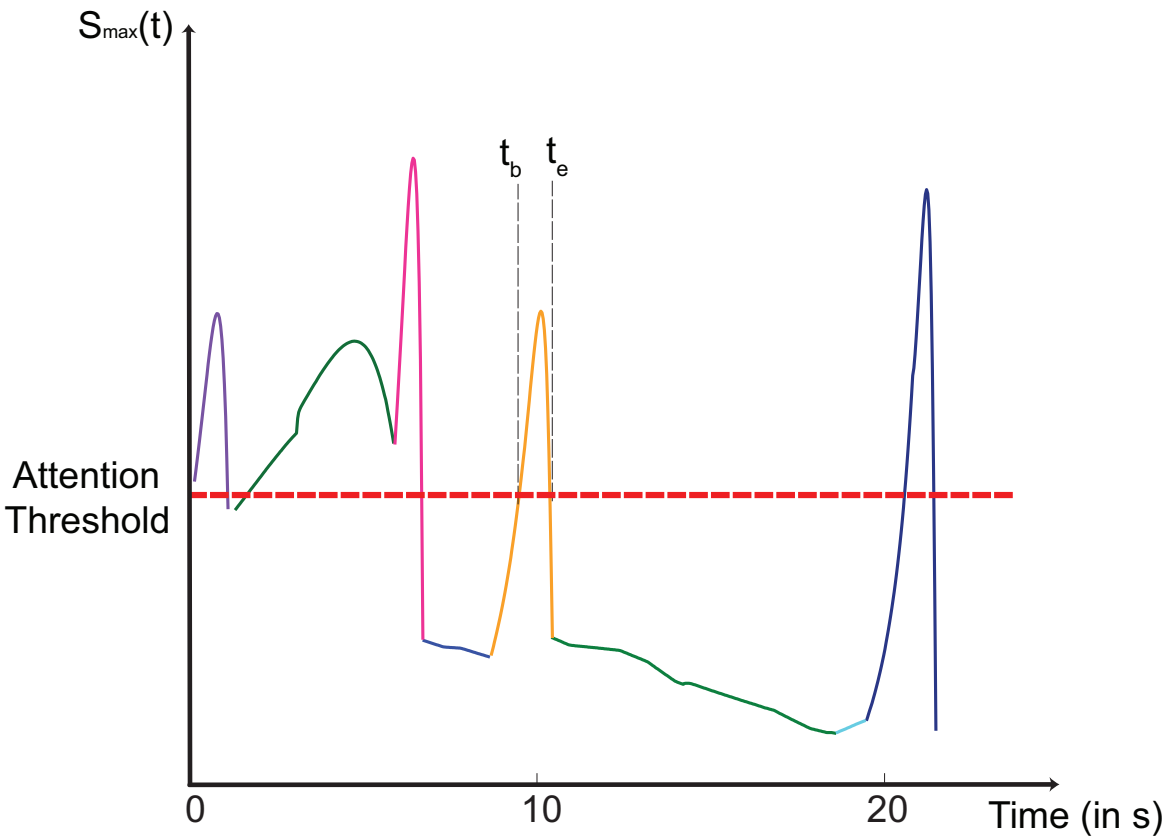


Figure 1: Overall maximum scores $S_{max}(t)$ for a character C. Different colors represent different interest points. t_b and t_e represent the beginning and the end of a look-at constraint.

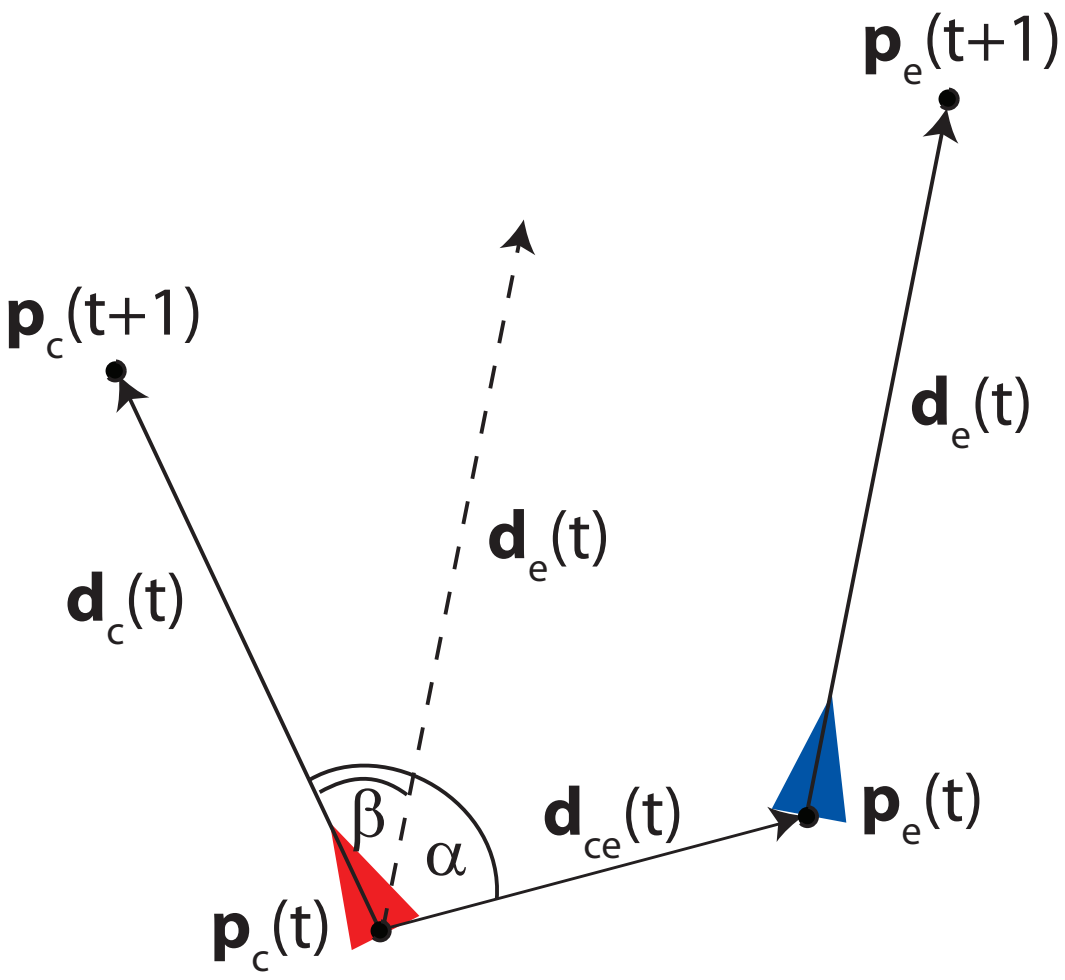


Figure 2: Schematical representation of the parameters used for the elementary scoring. $\mathbf{p}_c(t)$ is the character position at time t, $\mathbf{p}_e(t)$ is the entity position at time t, α is the entity orientation in the character's field of view, and β is the angle between the character and the entity forward directions.

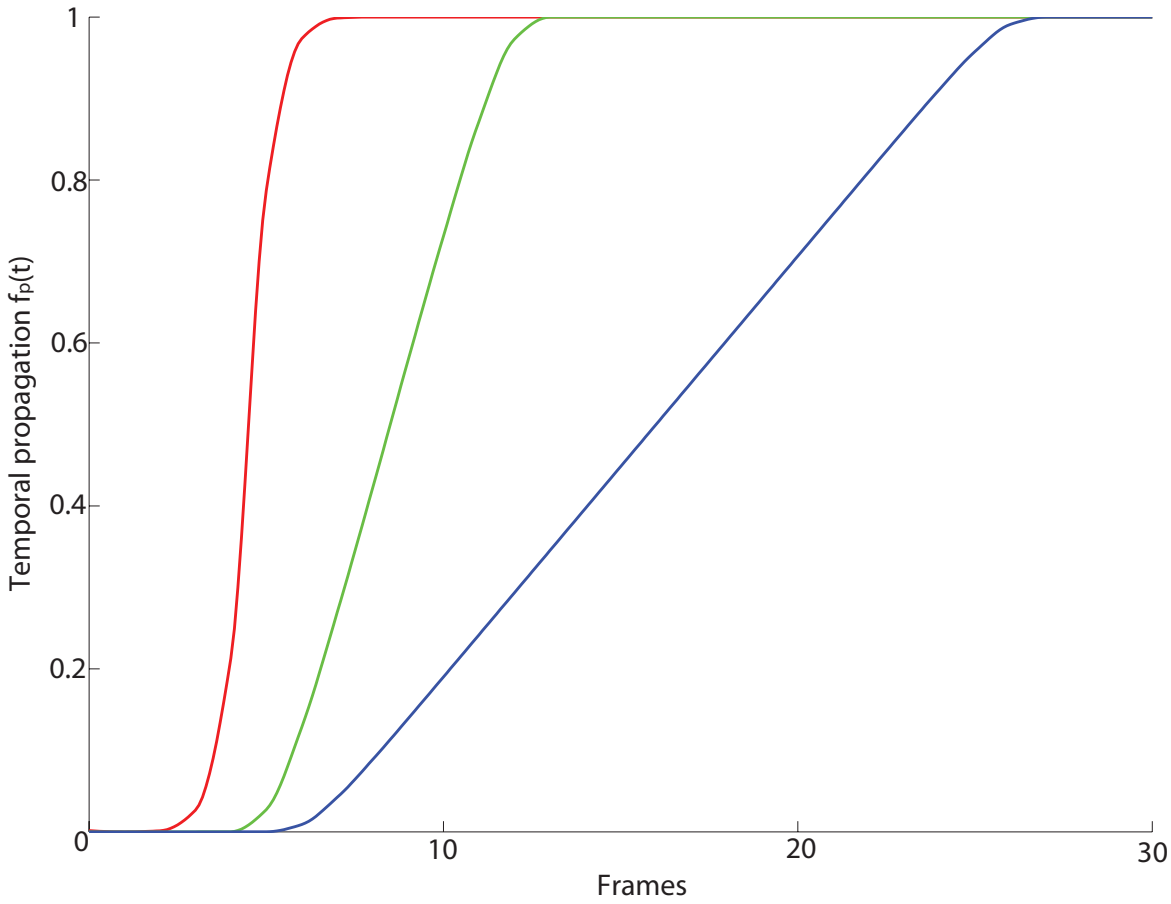


Figure 3: Desynchronization between the eyes, head, and torso. The eyes start moving before the head and satisfy the constraint first. The head and cervicals start moving and satisfy the constraint before the remainder of the spine.

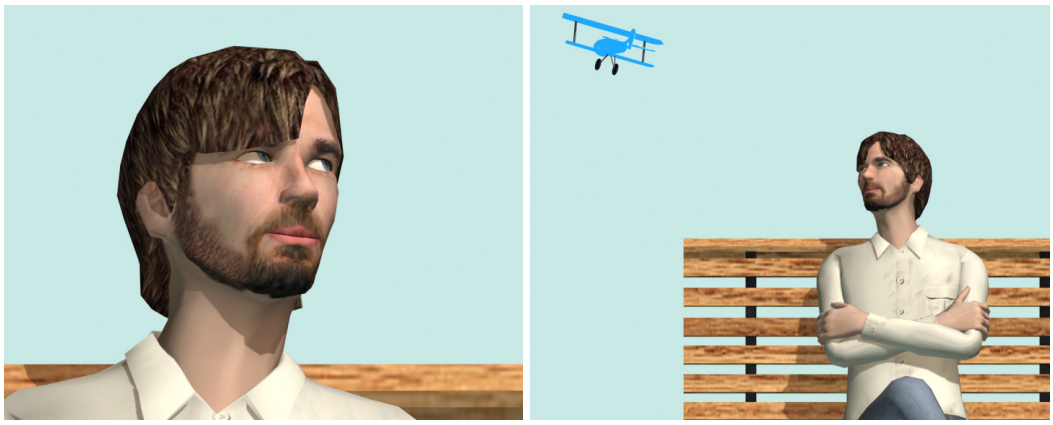
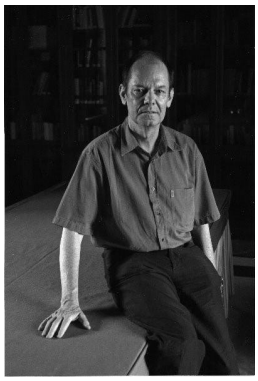


Figure 4: A character following an interest point with different sets of parameters. *Left:* Desynchronization between eyes, head, and torso. *Right:* Periphery parameter illustration.

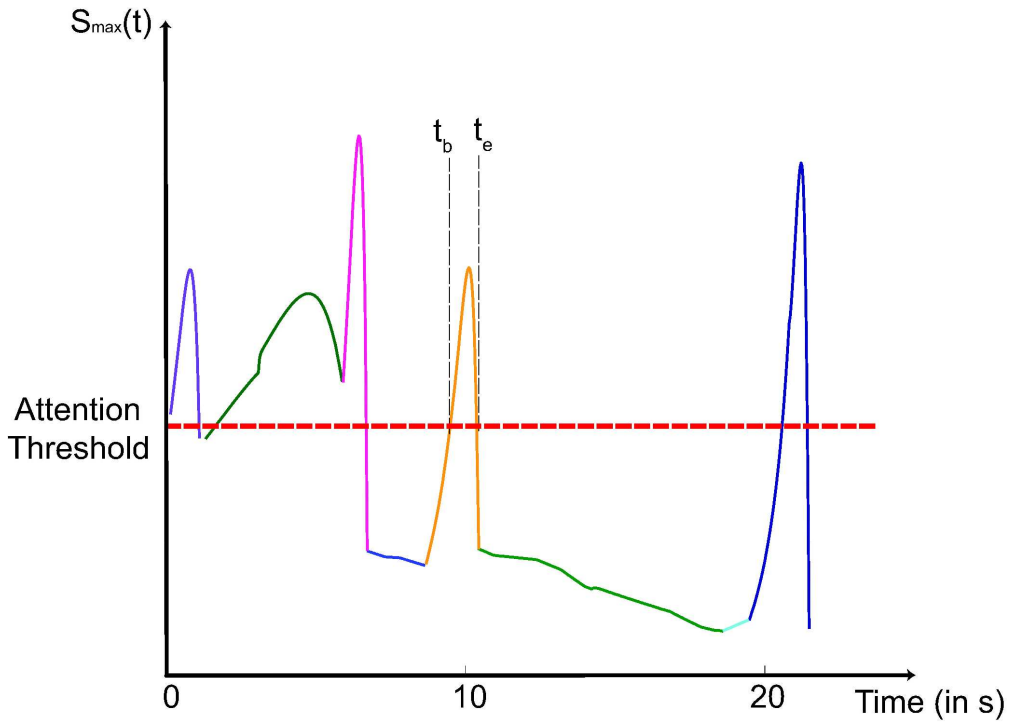
Figure 5: Examples of attention behaviors in a crowd animation.

Author Biographies

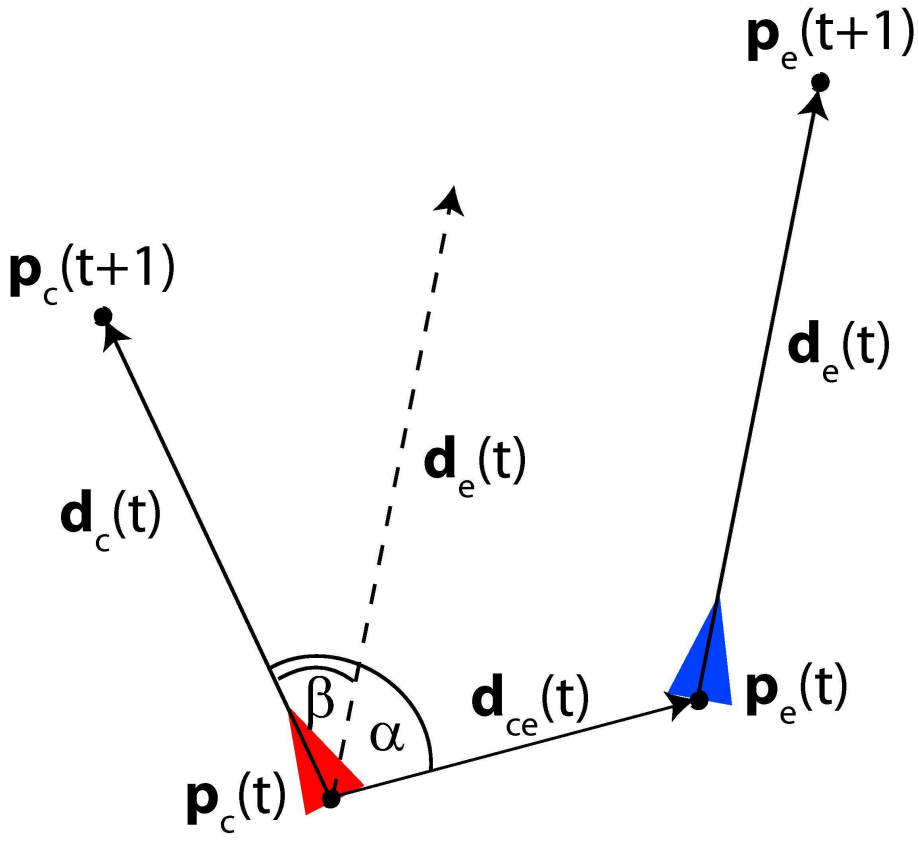
Helena Grillon is a Ph.D. candidate at the Virtual Reality Laboratory (VRLab), EPFL, Switzerland. She has obtained her bachelors degree in Information Systems and Communication at the Geneva University, Switzerland and her masters degree in Visualization and Graphics Communication at the EPFL. Her research interests are concentrated on character animation and behavior for use in Virtual Reality Exposure Therapies.



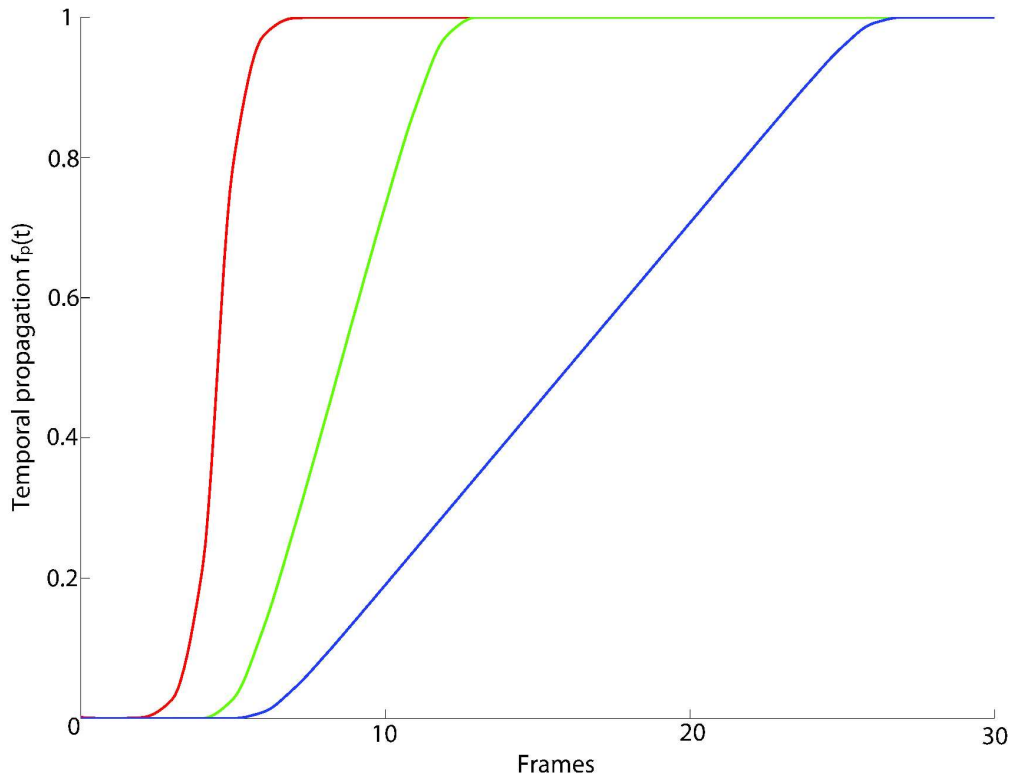
Daniel Thalmann is professor and director of The Virtual Reality Lab (VRlab) at EPFL, Switzerland. He is a pioneer in research on virtual humans. His current research interests include real-time virtual humans in Virtual Reality, crowd simulation, and multimedia virtual environments. He is coeditor-in-chief of the Journal of Computer Animation and Virtual Worlds and member of the editorial board of the Visual Computer and four other journals. Daniel Thalmann was member of numerous program committees, co-chair, and program co-chair of several conferences including IEEE VR, ACM VRST, CGI, SCA, CASA. He has also organized five courses at SIGGRAPH on human animation and crowd simulation. Daniel Thalmann has published numerous papers in graphics, animation, and virtual reality. He is coeditor of 30 books and coauthor of several books including ”Crowd Simulation” (2007). He received his Ph.D. in Computer Science in 1977 from the University of Geneva and an Honorary Doctorate (Honoris Causa) from University Paul- Sabatier in Toulouse, France, in 2003.



Overall maximum scores $S_{max}(t)$ for a character C. Different colors represent different interest points. t_b and t_e represent the beginning and the end of a look-at constraint.
263x207mm (600 x 600 DPI)



Schematical representation of the parameters used for the elementary scoring. $\mathbf{p}_c(t)$ is the character position at time t, $\mathbf{p}_e(t)$ is the entity position at time t, α is the entity orientation in the character's field of view, and β is the angle between the character and the entity forward directions.
106x92mm (600 x 600 DPI)



Desynchronization between the eyes, head, and torso. The eyes start moving before the head and satisfy the constraint first. The head and cervicals start moving and satisfy the constraint before the remainder of the spine.
287x222mm (600 x 600 DPI)

A character following an interest point with different sets of parameters.
Desynchronization between eyes, head, and torso.
254x203mm (600 x 600 DPI)

A character following an interest point with different sets of parameters.
Periphery parameter illustration.
254x203mm (600 x 600 DPI)

Examples of attention behaviors in a crowd animation.
450x343mm (600 x 600 DPI)

Examples of attention behaviors in a crowd animation.
450x343mm (600 x 600 DPI)

Examples of attention behaviors in a crowd animation.
450x343mm (600 x 600 DPI)